陕西师范大学

**python程序设计**

**课程设计报告**

**设计题目： “Learning Log（学习笔记）”**

**学 院： 计算机科学学院**

**专 业： 人工智能2301**

**学 号： 42312205**

**姓 名： 王特警**

**指导教师： 卢俊岭**

**2024 年 12 月 19 日**

# 

# 目录

[1. 设计目的和内容 2](#_Toc517097849)

[1.1 设计目的 2](#_Toc517097850)

[1.2 设计内容 2](#_Toc517097851)

[2. 设计思路 2](#_Toc517097852)

[3. 基本功能描述 3](#_Toc517097853)

[3.1 基本功能 3](#_Toc517097854)

[3.2 功能模块图 4](#_Toc517097855)

[4. 软件设计 5](#_Toc517097856)

[4.1 设计步骤 5](#_Toc517097857)

[4.2 系统开发工具 5](#_Toc517097858)

[4.3 数据格式 5](#_Toc517097859)

[4.4 结果展示 7](#_Toc517097860)

[4.5 关键功能 10](#_Toc517097861)

[5. 结论与心得体会 10](#_Toc517097862)

[6. 参考资料 11](#_Toc517097863)

[7. 附录 11](#_Toc517097864)

[7.1 调试报告 11](#_Toc517097865)

[7.2 测试结果 11](#_Toc517097866)

# 设计目的和内容

* 1. **设计目的**

网络上有各式各样、琳琅满目的学习笔记软件。他们有的功能丰富、有的特色鲜明。但是我想要建立一个极简主义的学习笔记。这个学习笔记没有冗余的功能，用户注册之后迎面而来的就是直观的学习笔记页面。

* 1. **设计内容**

一款极简主义的学习笔记，主要具有以下几大功能:

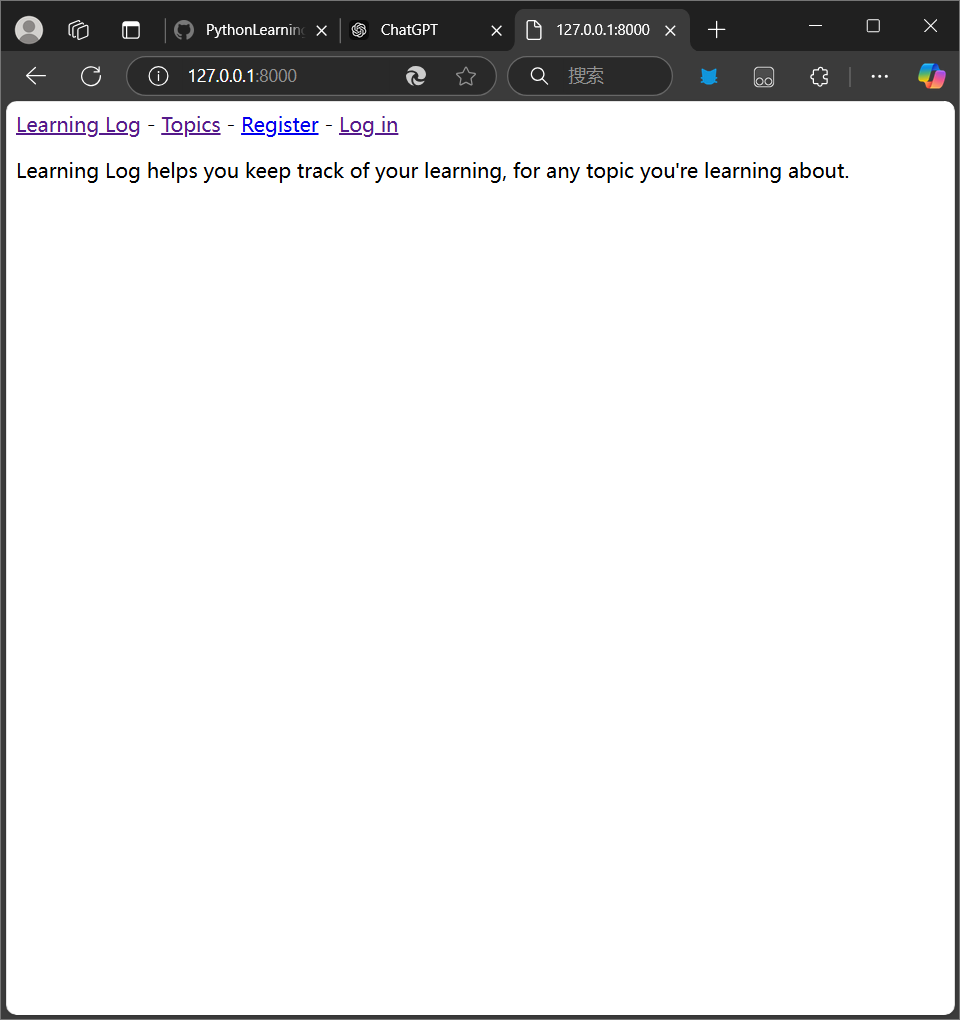
1. 用户注册和登录
2. 编写学习主题和在主题中编写学习内容
3. 用户注销

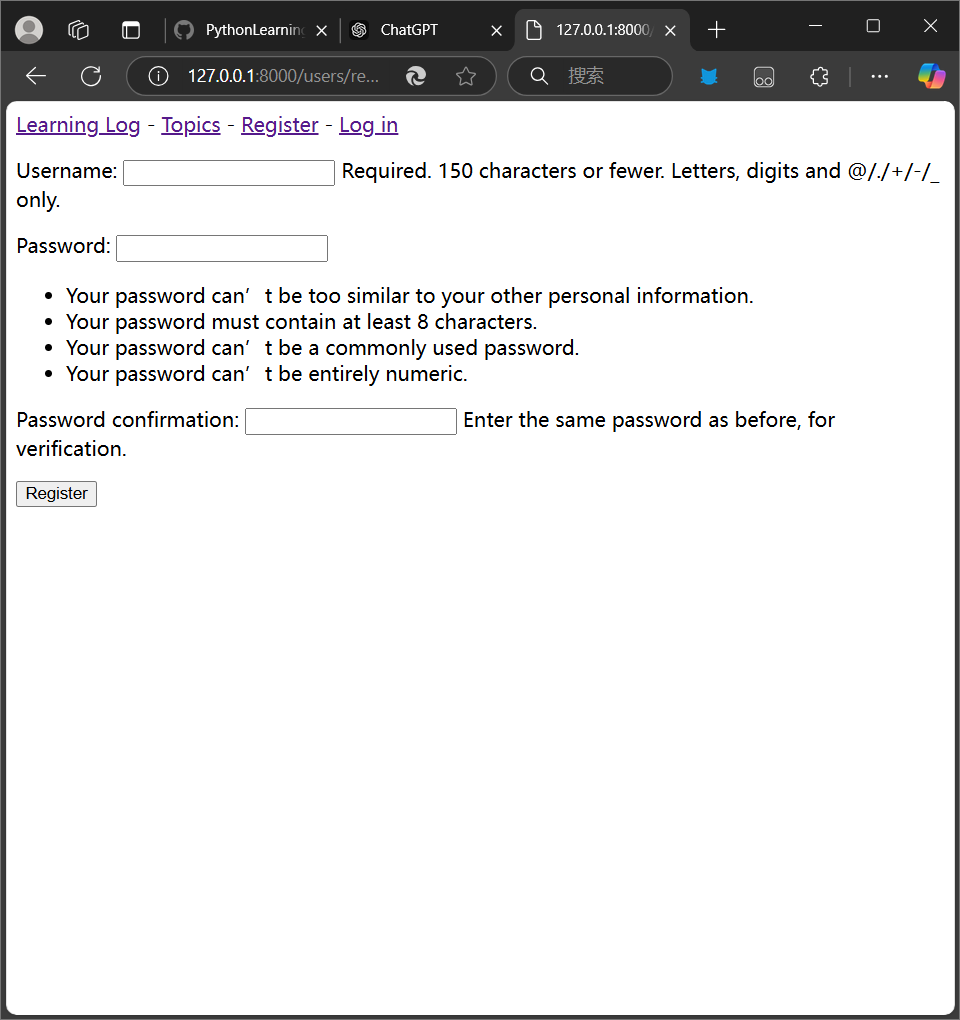
# 设计思路

1. 使用工具：  
   框架：django-5.1.4
2. 设计基本的学习笔记功能：编写主题与内容。先用Django开启一个主程序learning\_log，然后开启一个learning\_logs用来实现编写主题与内容。
3. 设计用户注册、登录以及注销功能：程序users用来完成这一功能。

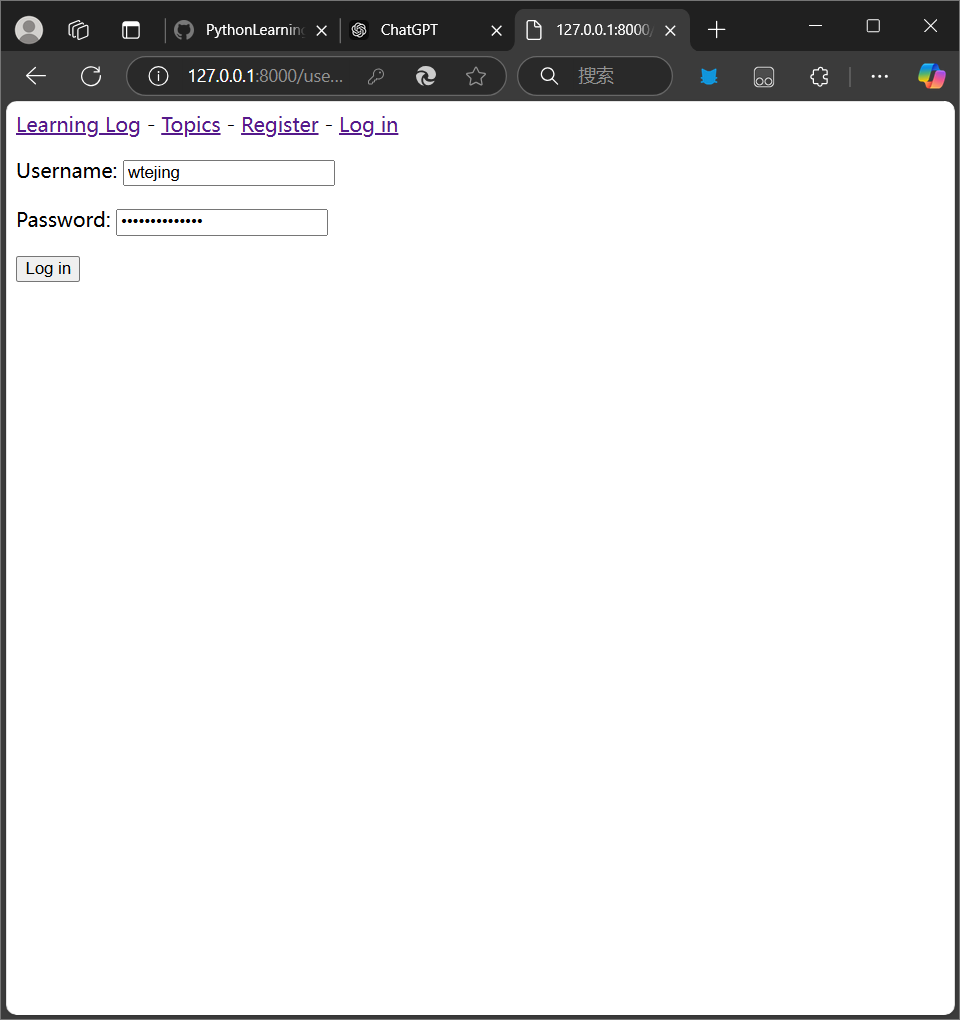
# 基本功能描述

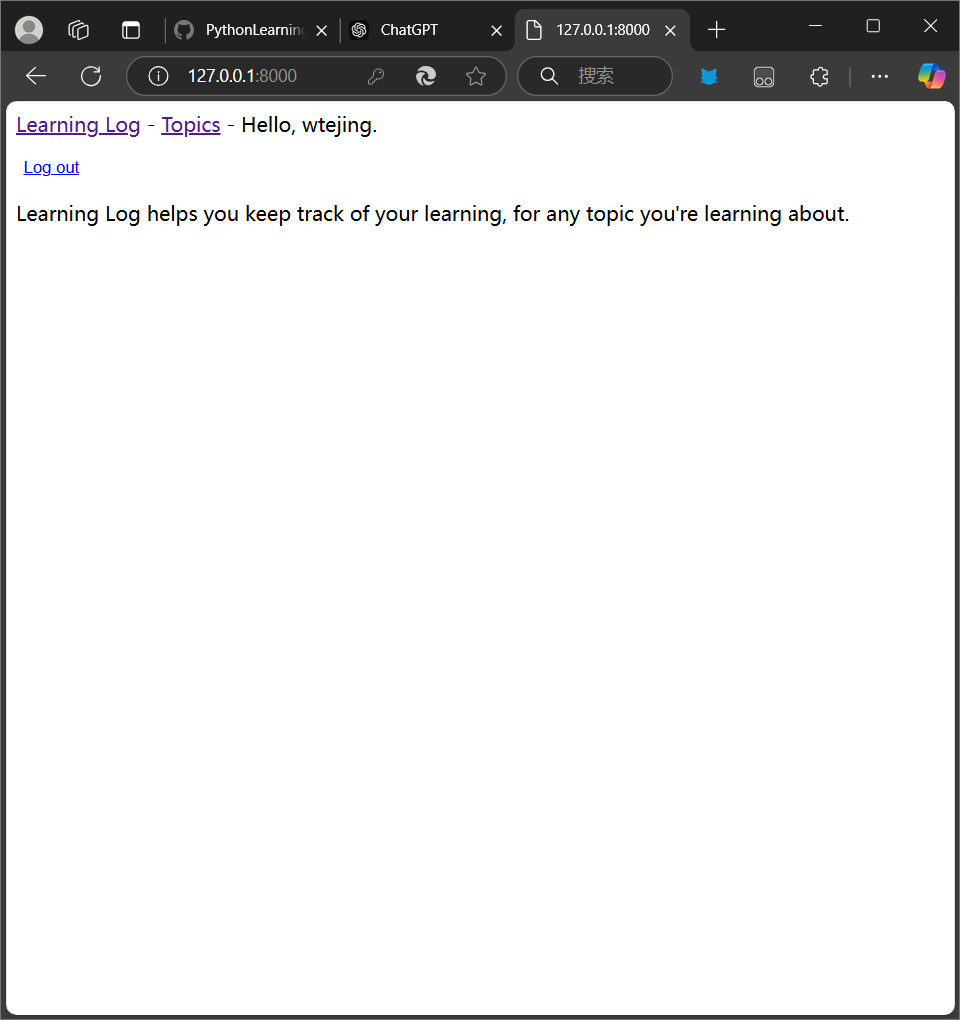
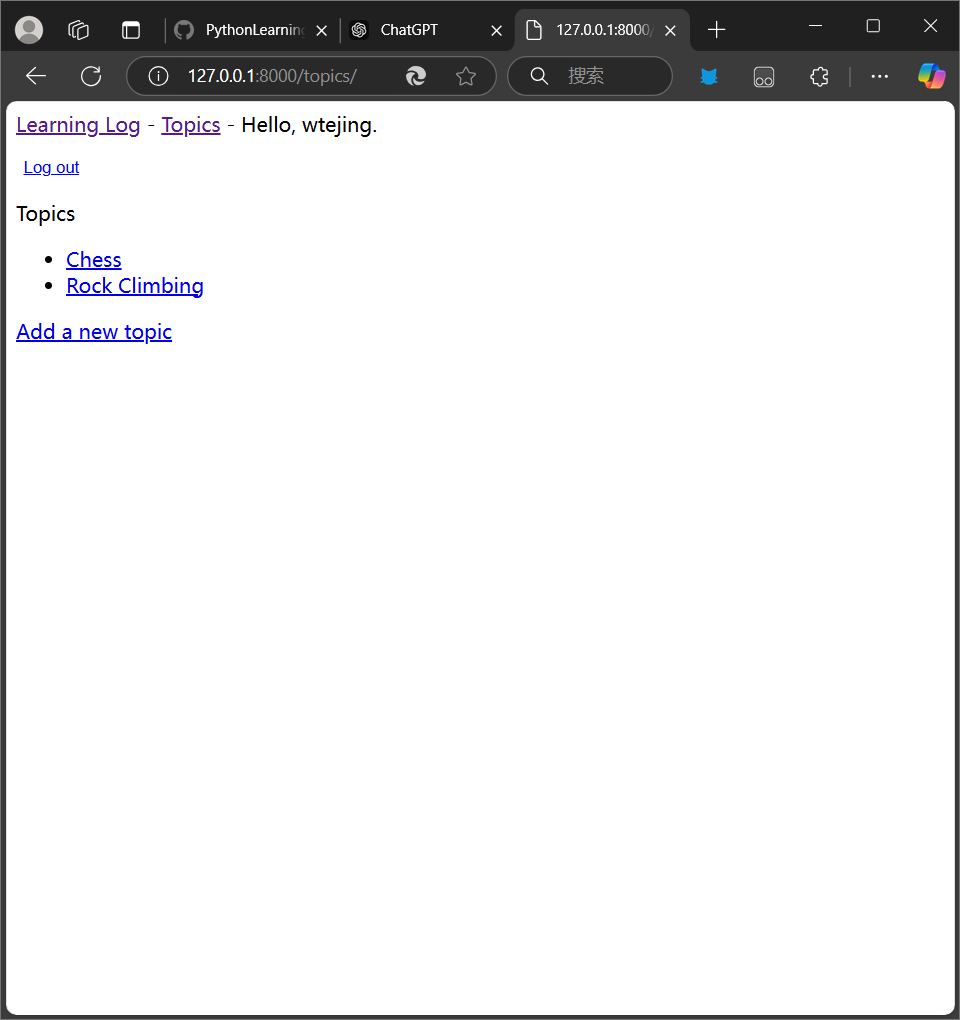
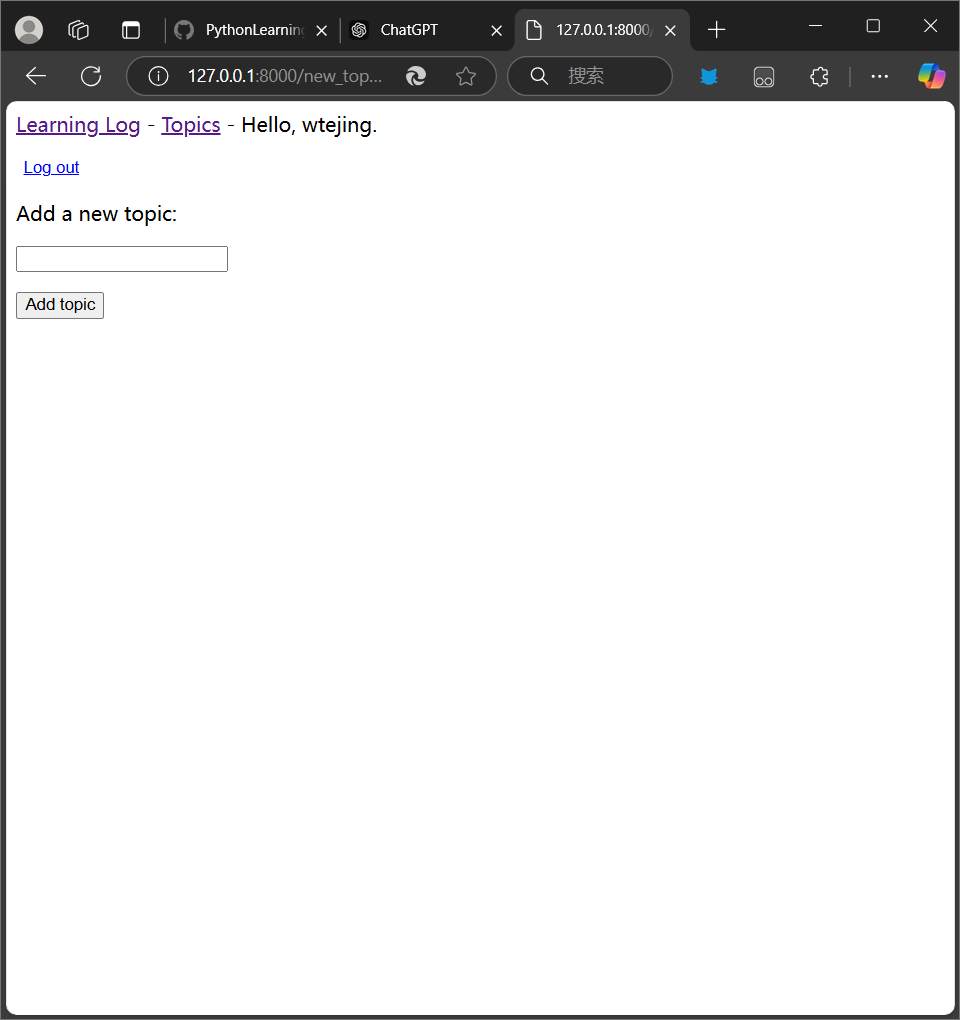
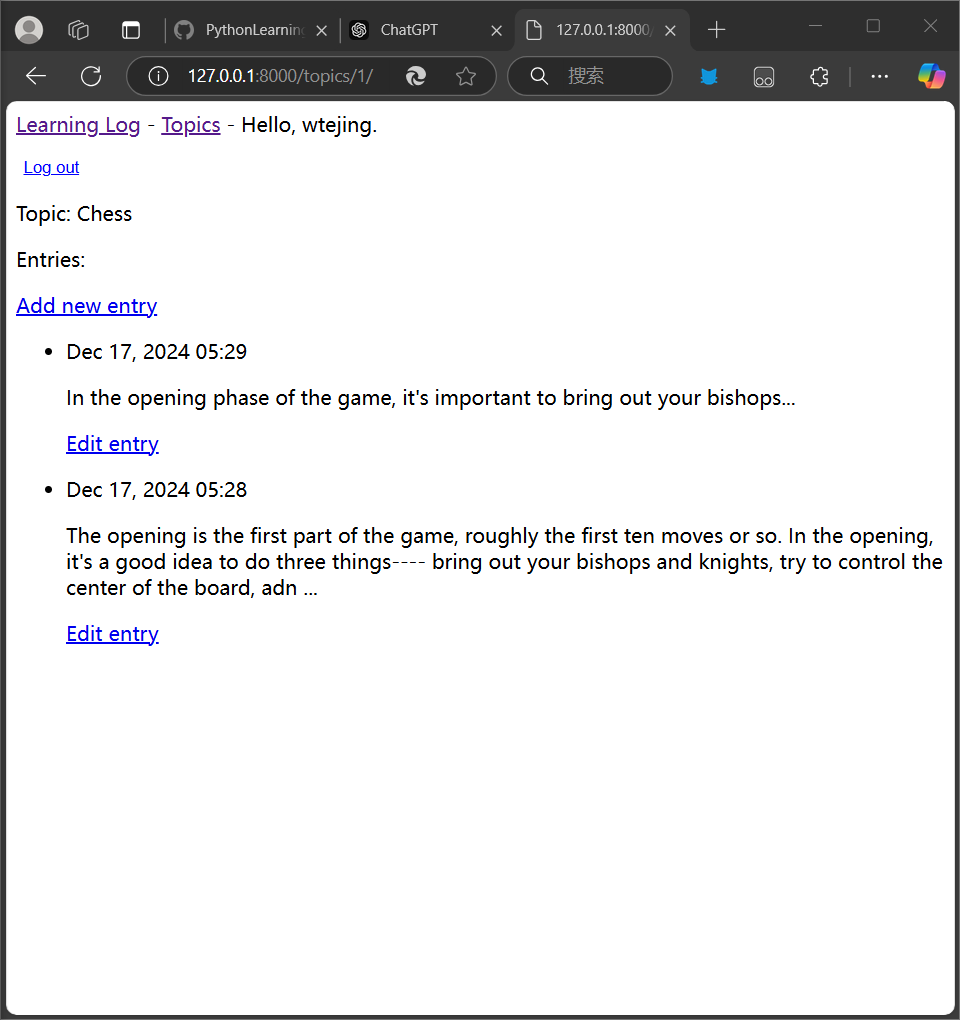
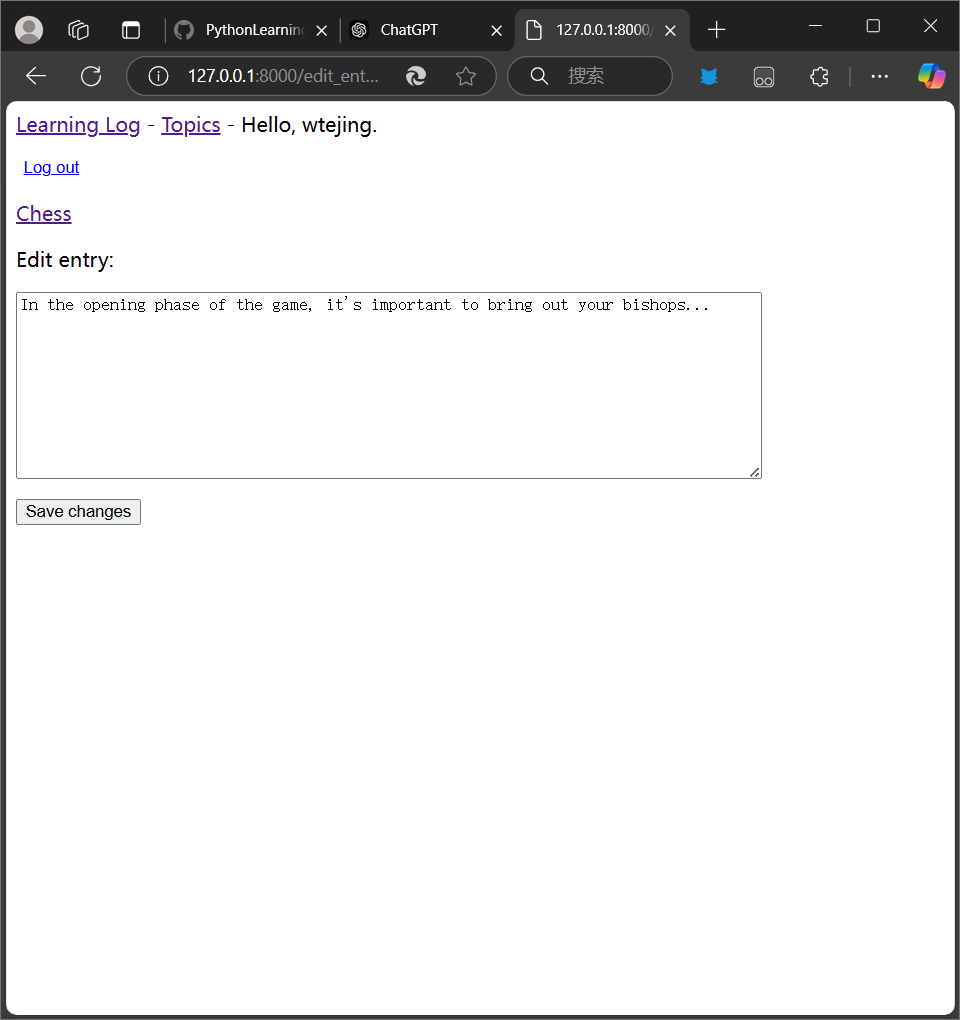
* 1. **基本功能**

1、主界面：  
 包括Learning Log, Topics, Register, Log in, 以及欢迎语：“Learning Log helps you keep track of your learning, for any topic you're learning about.”  
 其中，Learning Log链接到主页面，Topics已登陆的话链接到主题，未登录则链接至登陆页面。Register是注册，Log in是登录界面。  


2、注册界面  
 该界面用于新用户注册信息，包括用户名，二次输入密码。还附有一些关于账号密码的提醒信息。  


3、登录界面  
 输入已注册的账号密码登录。



1. 学习笔记主页面  
    主页面仍然包含Learning Log, Topics.  
    此外，还包含欢迎语“ Hello, wtejing.”  
    以及介绍语“Learning Log helps you keep track of your learning, for any topic you're learning about.”  
   
2. 主题大纲界面  
    点击Topics，进入主题页面，见下图，我已经创建了两个主题，分别是“Chess”和“Rock Climbing”。  
    下面还有一个选项“Add a new topic”，点击即可创建新主题。  
   
3. 新建主题  
    在框中输入主题名字，点击“Add topic”，即可添加新主题。  
   
4. 具体主题界面  
   点击某个主题，比如“Chess”。即可进入该主题的具体界面。如下，Chess主题包含两个学习笔记，每个笔记开头都有时间戳，并显示笔记的具体内容，每个笔记下面还有“Edit entry”，可以对笔记进行编辑。  
   
5. 编辑笔记  
    点击“Edit entry”即可进入该界面。在白框中输入笔记内容，点击“Save changes”即可保存。  
   
6. 注销  
   在主界面，用户点击“Log out”即可注销退出当前账户  
   1. **函数介绍**

|  |  |
| --- | --- |
| 函数名 | **功能** |
| edit\_entry(request, entry\_id): | **编辑新的学习条目** |
| new\_entry(request, topic\_id): | **新建学习条目** |
| new\_topic(request): | **新建学习主题** |
| topic(request, topic\_id): | **学习笔记主界面** |
| topics(request): | **展示主题** |
| register(request): | **注册界面** |

* 1. **功能结构**

Learning Log

注册登录

编辑笔记

增加主题

记录到数据库

登记到数据库

输入账号密码

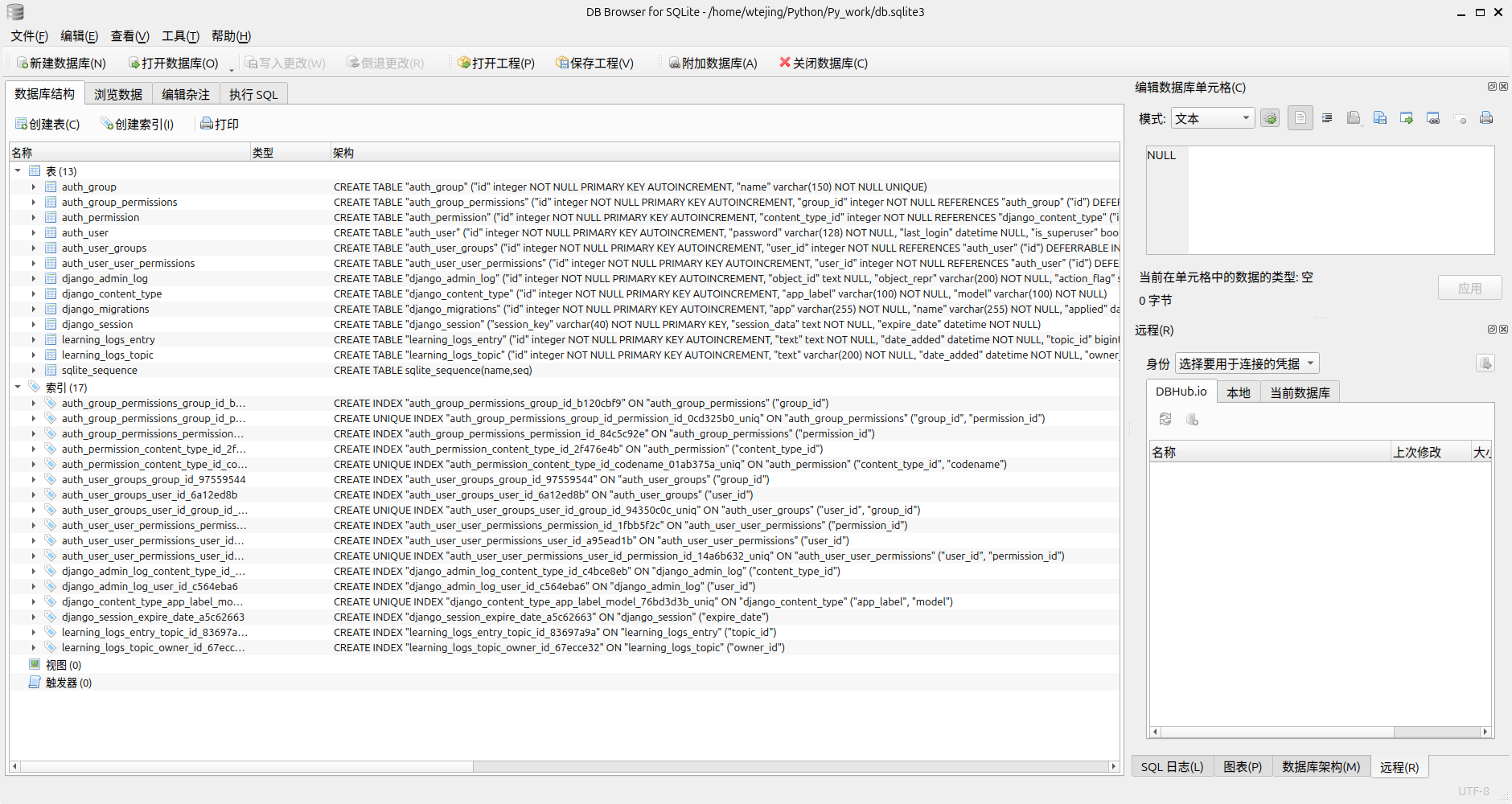
记录到数据库

# 软件设计

* 1. **设计步骤**

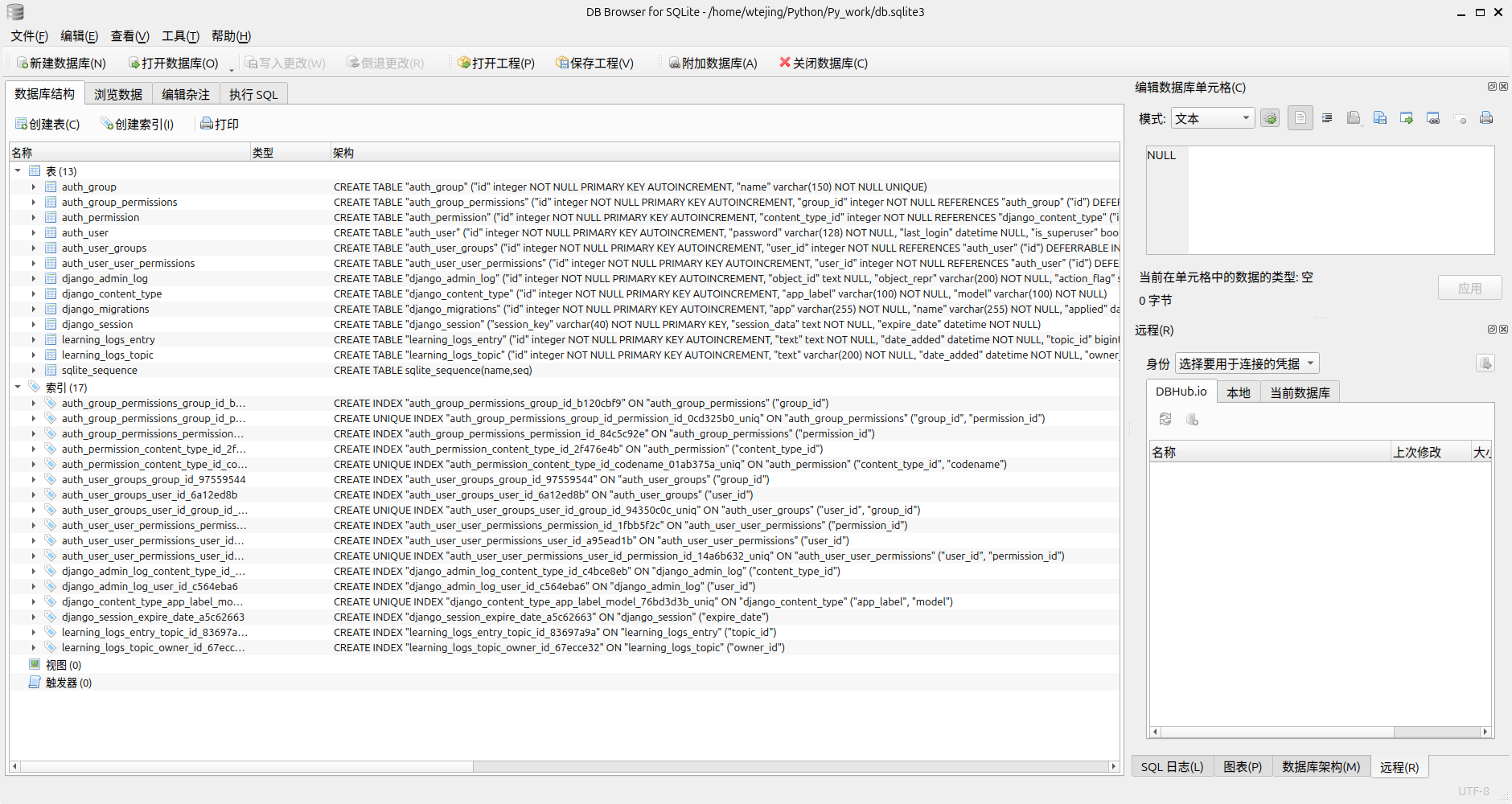
1. 调查用户需求，书写功能需求文档。
2. 查阅相关资料，学习用django的用法。
3. 设计数据库:数据库用来存储用户的账号密码以及学习笔记。
4. 设计用户界面，并加入各个功能控件。
5. 编写代码
6. 使用单元测试测试各个模块的功能。
7. 将所有功能模块连接起来，测试整个软件的功能，并不断调试。
   1. **系统开发工具**

系统环境：Ubuntu24.04 LTS  
Python版本：3.12  
编程工具：Ubuntu虚拟终端模拟器bash、vim

* 1. **数据格式**

用户的数据存储在本地的文本数据库中，使用DB Browser for SQlite即可查看

* 1. **结果展示**我将网站的使用效果录制成了一个视频，在视频中我实现了新用户的注册、登录、新建笔记等功能 ****
  2. **关键功能**1、由于本Web有多个界面，所以设计了一个父界面base.html：  
      <p>  
      <a href="{% url 'learning\_logs:index' %}">Learning Log</a> -  
      <a href="{% url 'learning\_logs:topics' %}">Topics</a> -  
      {% if user.is\_authenticated %}  
      Hello, {{ user.username }}.  
      <form action="{% url 'users:logout' %}" method="POST" style="display:inline;">  
      {% csrf\_token %}  
      <button type="submit" style="background:none;border:none;color:blue;text-decoration:underline;cursor:pointer;">  
      Log out  
      </button>  
      </form>  
      {% else %}  
      <a href="{% url 'users:register' %}">Register</a> -  
      <a href="{% url 'users:login' %}">Log in</a>  
      {% endif %}  
     </p>  
       
     {% block content %}{% endblock content %}  
     然后在其他界面代码的开头加上{% extends "learning\_logs/base.html" %}  
     这样，在其他界面就不需要写主界面已经有的代码了，便于网站的管理。

1. 规定用户可见的页面范围，在view.py视图文件中，先导入from django.contrib.auth.decorators import login\_required，然后在需要为特定用户展示的界面函数之前添加上@login\_required，如下：  
    from django.shortcuts import render, redirect  
   from django.contrib.auth.decorators import login\_required  
   from django.http import Http404  
     
   from .models import Topic, Entry  
   from .forms import TopicForm, EntryForm  
     
   # Create your views here.  
   def index(request):  
    """the main page of learning\_logs"""  
    return render(request, 'learning\_logs/index.html')  
     
   @login\_required  
   def topics(request):  
    """display all topics."""  
    topics = Topic.objects.filter(owner=request.user).order\_by('date\_added')  
    context = {'topics': topics}  
    return render(request, 'learning\_logs/topics.html', context)  
     
   @login\_required  
   def topic(request, topic\_id):  
    """display single topic and all items of it"""  
    topic = Topic.objects.get(id=topic\_id)  
    # 确认请求的主题属于当前用户  
    if topic.owner != request.user:  
    raise Http404  
    entries = topic.entry\_set.order\_by('-date\_added')  
    context = {'topic': topic, 'entries': entries}  
    return render(request, 'learning\_logs/topic.html', context)  
     
   @login\_required  
   def new\_topic(request):  
    """add new topic"""  
    if request.method != 'POST':  
    # didn't commit data: create a new from  
    form = TopicForm()  
    else:  
    # POST commits data: process data  
    form = TopicForm(data=request.POST)  
    if form.is\_valid():  
    new\_topic = form.save(commit=False)  
    new\_topic.owner = request.user  
    new\_topic.save()  
    return redirect('learning\_logs:topics')  
     
    # display empty form or indicate form is unavailable  
    context = {'form': form}  
    return render(request, 'learning\_logs/new\_topic.html', context)  
     
   @login\_required  
   def new\_entry(request, topic\_id):  
    """add new items in particular topic."""  
    topic = Topic.objects.get(id=topic\_id)  
     
    if request.method != 'POST':  
    # didn't commit data: create a empty form  
    form = EntryForm()  
    else:  
    # POST's data: process data  
    form = EntryForm(data=request.POST)  
    if form.is\_valid():  
    new\_entry = form.save(commit=False)  
    new\_entry.topic = topic  
    new\_entry.save()  
    return redirect('learning\_logs:topic', topic\_id=topic\_id)  
     
    # display empty form or indecate form data is unavailable  
    context = {'topic': topic, 'form': form}  
    return render(request, 'learning\_logs/new\_entry.html', context)  
     
   @login\_required  
   def edit\_entry(request, entry\_id):  
    """edit entries."""  
    entry = Entry.objects.get(id=entry\_id)  
    topic = entry.topic  
    if topic.owner != request.user:  
    raise Http404  
     
    if request.method != 'POST':  
    # 初次请求：使用当前条目填充表单.  
    form = EntryForm(instance=entry)  
    else:  
    # POST提交的数据：对数据进行处理  
    form = EntryForm(instance=entry, data=request.POST)  
    if form.is\_valid():  
    form.save()  
    return redirect('learning\_logs:topic', topic\_id=topic.id)  
     
    context = {'entry': entry, 'topic': topic, 'form': form}  
    return render(request, 'learning\_logs/edit\_entry.html', context)
2. 该程序使用了sqlite3来存储用户的账号和密码以及学习笔记，sqlite3的数据可以使用DB Browser for SQlite来可视化查看：  
   

# 5结论与心得体会

学习Django真是一次艰辛的旅程。刚开始接触Django真的是什么都不懂，看啥都看不会，一头雾水。真的感觉无力应对建设网站这个任务。但是慢慢的，熟悉了一些操作之后，感觉逐渐可以慢慢做起来了。先开始用Python manage.py startapp创建网站的新程序，然后添加新界面的url,设计界面的试图，写界面的模板......

我的学习笔记网站只有两个程序，一个learning\_logs，一个users。虽然功能不多，但我学会了很多。在完成这两个程序的简单部署之后，还会遇见其他的问题，比如如何规划一个独立用户的可见范围，如何避免受到对服务器的恶意攻击，等等，这都是要解决的问题。通过这次大作业，我对Django有了更深的理解，相信对我未来的生活也会更有帮助。

# 6参考资料 1、[最新Python的web开发全家桶（django+前端+数据库）\_哔哩哔哩\_bilibili](https://www.bilibili.com/video/BV1rT4y1v7uQ/?spm_id_from=333.337.search-card.all.click)

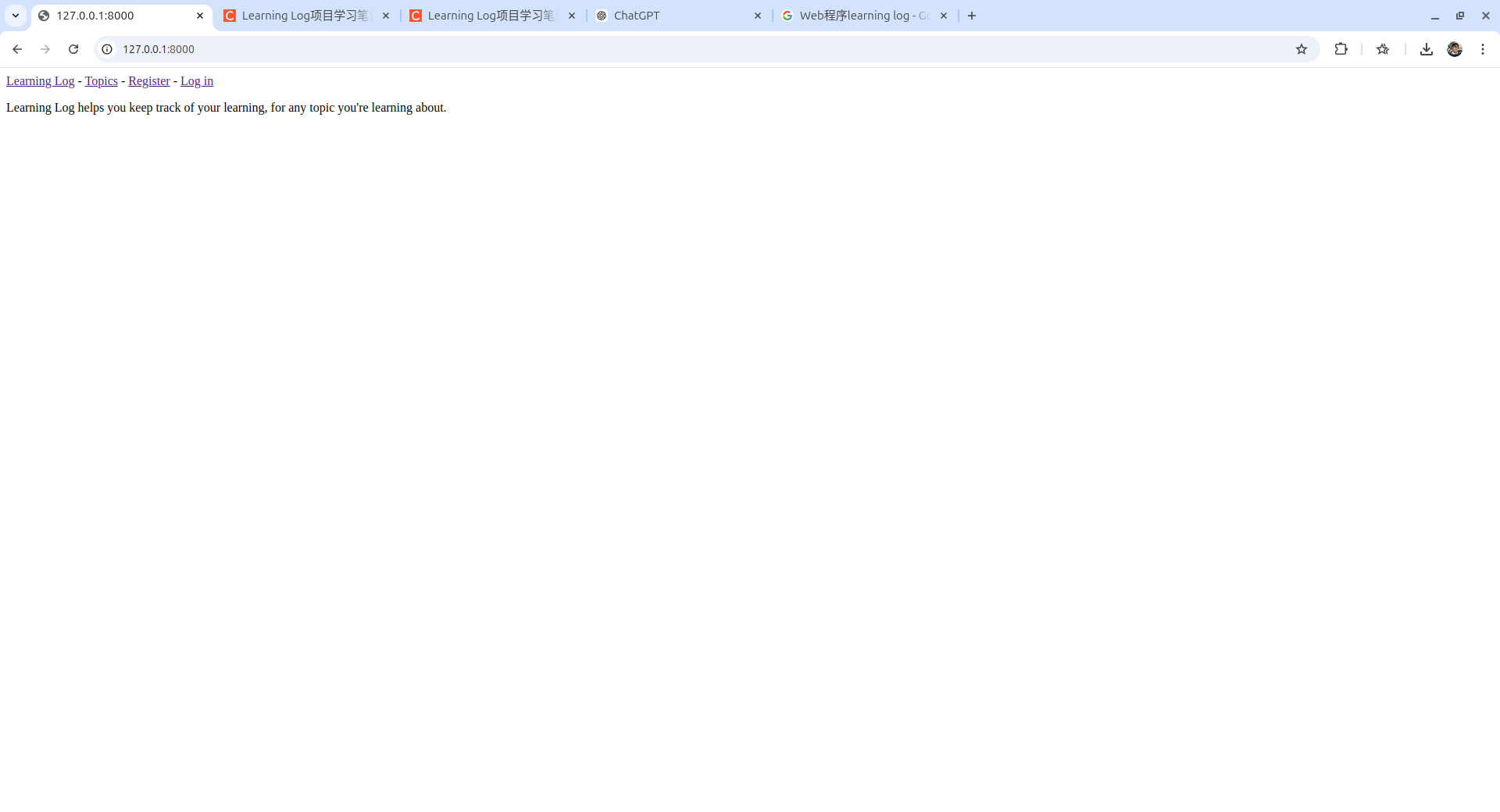
1. <https://blog.csdn.net/2301_79350630/article/details/132375488>
2. 《Python编程：从入门到实践》

# 7附录

**7.1调试报告 sss**

在调试程序过程中遇到了许多问题，其中最主要的一个就是主页面的Log out注销选项起初始终用不了，点击后总是显示“网页无法运行”，浏览了许多遍代码也没发现问题，最后才发现原来是 主界面代码base.html中关于Log out 的代码已经不被最新的django支持，更换为如下：  
 <button type="submit" style="background:none;border:none;color:blue;text-decoration :underline;cursor:pointer;">  
即可

**7.2测试结果**

  
修改代码后，运行结果正常，用户可正常返回起始页面。